第5章 字符串

字符串是 Python 语言中的基本数据类型，属于 Python 序列类型的一种。在数学中，序列也被称为数列，是指按照一定顺序排列的一列数。在程序设计中，序列是常用的数据结构，如 C、Java 等程序语言中的数组也是序列类型的一种。

为了实现某项功能，人们经常需要对字符串进行特殊处理，如拼接几个字符串、截取字符串、格式化字符串等。本章将重点介绍字符串类型以及相关的函数与方法。

5.1 任务一 数字和英文的对应

英文字母有 26 个，如果输入 1～26 之间的数字，如何得到对应的英文字母呢？根据 26 个英文字母在字母表中的位置，得到对应关系为：A1 B2 C3 D4 E5 F6 G7 H8 I9 J10 K11 L12 M13 N14 O15 P16 Q17 R18 S19 T20 U21 V22 W23 X24 Y25 Z26。

5.1.1 任务目标

编写程序：从键盘输入一个数字，输出对应的英文字母。

5.1.2 操作步骤

（1）在 IDLE 中创建新文件，输入代码，如程序段 5-1 所示。

程序段 5-1

```
s="ABCDEFGHIJKLMNOPQRSTUVWXYZ"
n=input()
print s[n-1]
```

（2）运行程序，结果如图 5-1 所示。

5.1.3 必备知识

5.1.3.1 字符串数据类型

字符串作为 Python 语言的简单数据类型，实际就是由数字、字母、下画线、特殊符

```
========== RESTART: D:/Python/p5-1.py ==========
1
A
>>>
========== RESTART: D:/Python/p5-1.py ==========
12
L
>>>
========== RESTART: D:/Python/p5-1.py ==========
26
Z
```

图 5-1　任务一运行结果

号、各国文字等组成的连续字符序列。关于字符串的几点说明如下。

(1) 引号括起来的都是字符串。

用引号括起来的都是字符串。引号可以是单引号、双引号、三单或三双引号，这三种形式没有语义上的区别，其中单引号和双引号的字符串必须在同一行上，而三引号的字符串可以分布在连续的多行上。

单引号字符串，例如：

```
'MonTueWedThuFriSatSun'
```

双引号字符串，例如：

```
"Happy is simple just like 123"
```

三引号括起来的多行字符串，例如：

```
''' In order to learn Python, you need to do:
      'Read book', "Write Program", `Debug Program' '''
```

在由三单引号括起来的第三个字符串中，里面包括了换行，单引号，双引号等。同理，也可以把三单引号换成三双引号。

(2) 通过“=”，可以实现字符串的赋值。

例如，将字符串'MonTueWedThuFriSatSun'赋值给变量 Week。

```
Week='MonTueWedThuFriSatSun'
```

(3) 用 type()函数，可查看变量类型。

type()函数可以查看变量的类型，<'str'>即为字符串类型，str 是单词 string 的缩写。type()函数的使用如图 5-2 所示。

```
>>> Week='MonTueWedThuFriSatSun'
>>> print(type(Week))
<type 'str'>
```

图 5-2　type()函数示例

(4) 字符串的输入使用 raw_input()函数,语法格式为:

```
变量名=raw_input([提示信息])
```

raw_input()语句接收来自键盘输入的字符串,将其赋值给字符串类型的变量。

(5) 字符串的输出使用 print 语句。

可以直接用 print 语句输出字符串,也可以通过%s 格式控制符来输出。例如下面的代码:

```
S1=raw_input()              #接收来自键盘输入的字符串
print S1                    #直接输出字符串
print "%s"%S1               #格式控制符输出字符串
```

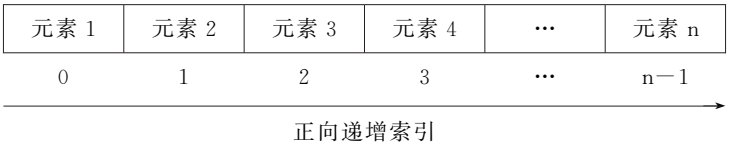
5.1.3.2 字符串的索引

字符串存储于连续的内存空间中。内存空间按一定的顺序存放字符串的每个值。每一个值(即元素)都被分配一个数字,该数字称为索引或位置,通过它,可以得到对应元素的值。

Python 提供两种索引方式,正向递增索引与反向递减索引。

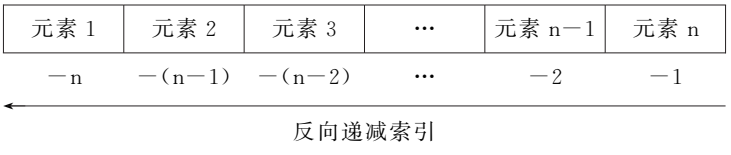
(1) 正向递增索引。

索引从 0 开始,由左向右,正向递增。下标为 0,表示第一个元素;下标为 1,表示第二个元素;以此类推,最大下标是字符串长度(n)减 1。



(2) 反向递减索引。

为了避免与正向索引的第一个元素下标 0 重合,反向索引的索引值从−1 开始,从右向左依次递减。



通过索引可以访问字符串中的任何元素。语法格式为:

```
<字符串变量名>[索引值]
```

例如下面的代码,输出字符串 S1 的第一个元素 I 与倒数第二个元素 h,分别通过访问下标为 0 和下标为−2 的元素来实现,如图 5-3 所示。

```
>>> S1='I love Python very much!'
>>> print S1[0]
I
>>> print S1[-2]
h
>>> print S1[24]

Traceback (most recent call last):
  File "<pyshell#14>", line 1, in <module>
    print S1[24]
IndexError: string index out of range
```

图 5-3　索引示例

一定要注意下标的索引范围，正向是从 0 开始，反向是从－1 开始，不能超出字符串的最大长度。如果字符串 S1 的字符总长度是 24 个，最后一个元素的下标为 23，如果访问 S1[24]的元素，就会引发索引错误——IndexError，该错误的含义是，超出了字符串的索引下标范围。

5.2　任务二　身份证信息解析

我国居民身份证号码由 18 位数字组成，处于每个不同位置的数字代表着不同的含义。以身份证号码 32010519820927512X 为例：

前 1、2 位数字代表所在省（直辖市、自治区）的代码；

第 3、4 位数字代表所在地级市（自治州）的代码；

第 5、6 位数字代表所在区（县、自治县、县级市）的代码；

第 7～14 位数字代表出生年、月、日；

第 15、16 位数字代表所在地的派出所的代码；

第 17 位数字代表性别，奇数表示男性，偶数代表女性；

第 18 位数字是校验码，用来检验身份证的正确性，它是通过既定的公式推算得到的。校检码可以是 0～9 之间的数字，如果推算出来的数字为 10，则用 X 来表示。

5.2.1　任务目标

编写程序：从键盘输入一个 18 位居民身份证号码，输出该居民的出生日期、性别、校验码信息。

5.2.2　操作步骤

（1）在 IDLE 中创建新文件，输入代码，如程序段 5-2 所示。

程序段　5-2

```
s=raw_input()                    #输入身份证号码
```

```
year=s[6:10]                #提取出生年
month=s[10:12]              #提取出生月
day=s[12:14]                #提取出生日
gender=s[-2]                #提取性别
check=s[-1]                 #提取校验码
print "birthday: %s-%s-%s"%(year,month,day)
print "gender: %s"%gender
print "check: %s"%check
```

(2) 运行程序,结果如图 5-4 所示。

```
========================== RESTART: D:\Python\p5-2.py ==========================
32010519820927512X
birthday: 1982-09-27
gender: 2
check: X
```

图 5-4　任务二运行结果

5.2.3　必备知识

5.2.3.1　字符串的切片

字符串中某个子串或区间的检索被称为切片。切片操作非常方便,用户可以通过切片访问序列中一定范围内的元素,并生成一个新的序列。

切片的语法格式为:

```
Str[start:end:step]
```

说明如下。

(1) Str 代表字符串或者字符串变量名。

(2) start 表示切片的开始位置(包括该位置),如果不指定,就默认为 0。

(3) end 表示切片截止的位置(不包括该位置),如果不指定,就默认为包含序列的最后一个元素。

(4)step 表示切片的步长,默认为 1,省略步长时,最后一个冒号也省略。如果指定了步长,那么将按照该步长取得序列中对应的元素。

如果只保留 start 和 end 参数中间的冒号,start,end,step 三个参数都省略,则相当于复制整个序列,语法形式为:Str[:]。

程序段 5-2 就是利用字符串的切片操作,从身份证号码中提取不同位置的数字:

```
year=s[6:10]                #提取出生年
month=s[10:12]              #提取出生月
day=s[12:14]                #提取出生日
gender=s[-2]                #提取性别
check=s[-1]                 #提取校验码
```

5.2.3.2 利用切片逆序输出字符串

利用切片，可以非常方便地完成字符串的逆序输出，如 Str[::-1]可以直接实现字符串的逆序输出。例如下面代码：

```
S="Hello Python"
Print S[::-1]
```

5.3 任务三 输出图案

5.3.1 任务目标

输出由“*”图案构成的直角三角形，如图 5-5 所示，行数由键盘输入。

```
*
**
***
****
```

图 5-5 直角三角形图案

5.3.2 操作步骤

（1）在 IDLE 中创建新文件，输入代码，如程序段 5-3 所示。

程序段 5-3

```
n=input()
for i in range(1,n+1):
    print '* '*i
```

（2）运行程序，结果如图 5-6 所示。

```
========================== RESTART: D:/Python/p5-3.py ======
5
*
**
***
****
*****
>>>
========================== RESTART: D:/Python/p5-3.py ======
3
*
**
***
```

图 5-6 任务三运行结果

5.3.3 必备知识

字符串的内置基本运算符“+”与“*”的含义如表 5-1 所示。

表 5-1 字符串的内置基本运算符

运算符	含　义
+	连接字符串,得到新的字符串
*	将字符串重复若干次,生成新的字符串

5.3.3.1 字符串的拼接

Python 语言支持多个相同类型的序列的拼接操作,通过“加”(+)来实现,在拼接过程中不会去除重复的元素。

两个字符串 S1 和 S2 和一个字符“,”,用“+”运算符连接,可以组成一个新的字符串。例如下面代码:

```
S1="hello"
S2="python"
print S1+","+S2
```

类型不匹配的变量不能做“+”运算,例如下面的代码段中,s 是字符串,num 是数字,拼接时会产生类型错误,弹出错误提示“TypeError:Can not concatenate 'str' and 'int' objects”(提示:不能对字符串与整数进行拼接)。

```
s="My score is"
num=100
print s+num
```

为了修正该错误,可以将 print 语句修改为 print(s+str(num)),相当于将 num 转换成字符串之后再拼接,这样就不会产生类型错误了。

5.3.3.2 字符串的复制

通过乘法运算“*”,可以实现字符串的复制。

任务三中,每一行需要输出的“*”的个数相当于行数,利用字符的复制功能可以实现每一行字符的输出。

5.4 任务四　查找元音字母

5.4.1 任务目标

编写程序:从键盘输入字符串,统计字符串中元音字母的个数。元音字母包括 aeiou

及其大写字母或 AEIOU。

5.4.2 操作步骤

(1) 在 IDLE 中创建新文件,输入代码,如程序段 5-4 所示。

程序段 5-4

```
s=raw_input()
cnt=0
for c in s:
    if c in 'AEIOUaeiou':
        cnt=cnt+1
print cnt
```

(2) 运行程序,结果如图 5-7 所示。

```
========================== RESTART: D:/Python/p5-4.py =====
Life is short,you need python!
9
>>>
========================== RESTART: D:/Python/p5-4.py =====
Why learn python?
3
```

图 5-7 任务四运行结果

5.4.3 必备知识

5.4.3.1 字符串的判断运算符

字符串的内置判断运算符及其含义如表 5-2 所示。

表 5-2 字符串的判断运算符及其含义

运算符	含 义
in	判断字符串中是否包含某个字符串
==	判断字符串内容是否相同

1. in 运算符

通过 in 来判断,判断字符串中是否包含某个字符串,也可以用 not in 来判断是否不包含某个字符串。例如下面代码:

```
print "py" in "python"          #结果为 True
print "py" not in "python"      #结果为 False
```

2. ==运算符

“==”可以判断字符串的一致性。如果两个字符串相等,则输出 True(真),否则输出 False(假)。也可以通过“!=”来判断两个字符串的不一致。例如下面代码:

```
print "hello"=="hallo"          #结果为 False
print "hello" !="hallo"         #结果为 True
```

5.4.3.2 字符串的遍历

在任务四中,查找元音字母的算法思想是:将字符串 s 作为 for 循环结构的迭代器,通过遍历字符串 s 中的每个字符,依次用 in 判断每个字符是否在元音字符列表中。

字符串遍历常用的两种方法是 for in 遍历和下标法。

1. for in 遍历

for in 遍历适合对字符进行直接处理的场景。例如下面代码:

```
s=raw_input()
for c in s:
    print c,                    #逐一输出字符串中的每个字符,字符之间空格隔开
```

2. 下标法

下标法是用 range()函数,将字符串长度传入遍历字符串。range()函数适用于需要对字符的索引值进行判断的场景。例如下面代码:

```
s=raw_input()
for c in range(0,len(s),2):
    print c,                  #输出字符串中索引值为偶数的字符
```

3. 知识拓展

from _ _future_ _ import print_function 语句的作用是:允许在 Python 2 版本中使用 Python 3 的输出 print 功能。注意:future 关键字的前后各有两个下画线。

例如下面代码中,遍历输出字符串 s 的每一个字符,字符之间用逗号分隔,输出结果为 H,e,l,l,o。

```
from _ _future_ _ import print_function
s="Hello"
for c in s:
    print(c,end=",")
```

print 函数中用到了 end 关键字,指定每个字母之间的分隔符。

5.5 任务五 最大字符和最小字符

字符串的升序排序指从字符串的第一个字符开始比较，如果相等，就比较后一个；如果不等，就将较小的那一个放在较大的前面，这里的“大小”指的是字符的 ASCII 码值大小。

5.5.1 任务目标

输入字符串，完成以下任务：

（1）输出字符串的长度、最大字符和最小字符；

（2）将字符串升序排序并输出；

（3）将字符串逆序输出。

5.5.2 操作步骤

（1）在 IDLE 中创建新文件，输入代码，如程序段 5-5 所示。

程序段 5-5

```
s=raw_input()
slen=len(s)
print "The length of string is----",slen

smax=max(s)
print "The Max of string is----",smax

smin=min(s)
print "The Min of string is----",smin

s1=sorted(s)
print "The sorted string is----",s1

s2=reversed(s)
s2="".join(s2)
print "The inverse sorted string is----",s2
```

（2）运行程序，结果如图 5-8 所示。最大字符是“y”，最小字符是“ ”空格。

```
========================== RESTART: D:/Python/p5-5.py ===============
Why Learn Python?
The length of string is---- 17
The Max of string is---- y
The Min of string is----
The sorted string is---- [' ', ' ', '?', 'L', 'P', 'W', 'a', 'e', 'h', 'h', 'n', 'n', 'o', 'r', 't', 'y', 'y']
The inverse sorted string is---- ?nohtyP nraeL yhW
```

图 5-8 任务五运行结果